FOOD AND ECOLOGICAL SYSTEMS MODELLING JOURNAL

Food and Ecological Systems Modelling Journal 5: e122467 (2024)
DOI: https://doi.org/10.3897/fmj.5.122467

Methods

A general subpopulation model for the Animal Landscape and Man Simulation System (ALMaSS)

Xiaodong Duan[1], Christopher J. Topping[1]

1 *Social-Ecological Systems Simulation Centre, Department of Agroecology, Aarhus University, 8000 Aarhus, Denmark*
Corresponding author: Xiaodong Duan (xd@agro.au.dk)

Abstract

ALMaSS (the Animal Landscape and Man Simulation System) is a well-established modelling system known for its detailed agent-based animal models. However, for certain species, such as aphids, extremely high population sizes and limited detailed individual behaviour information make agent-based approaches either computationally expensive or unfeasible. Additionally, when these species are modelled as a prey base for other species, such as aphids for ladybirds, an efficient and accurate modelling approach is required. To address these challenges, we introduce a subpopulation-based model framework that effectively handles high population densities, while reducing computational complexity. This framework divides the landscape into a non-overlapping grid and population dynamics are managed within each grid using a flexible stage-structured population model. Grids are interconnected through short- and long-distance dispersal mechanisms, allowing for realistic movement and interactions. Parallel programming is employed to enhance performance, enabling faster simulation runs. We demonstrate the application of this model framework with a hypothetical species (not a species in the real world) and showcase its use in modelling aphid populations. This foundational subpopulation model could be used to model species with large population sizes within the ALMaSS framework and can be integrated into other simulation systems.

Key words: ALMaSS, dispersal, landscape simulation, mortality, reproduction, subpopulation modelling

OPEN ACCESS

Academic editor: Francesco Nazzi
Received: 6 March 2024
Accepted: 15 October 2024
Published: 12 November 2024

Citation: Duan X, Topping CJ (2024) A general subpopulation model for the Animal Landscape and Man Simulation System (ALMaSS). Food and Ecological Systems Modelling Journal 5: e122467. https://doi.org/10.3897/fmj.5.122467

Copyright: ©
Xiaodong Duan & Christopher J. Topping.
This is an open access article distributed under terms of the Creative Commons Attribution License (Attribution 4.0 International – CC BY 4.0).

Introduction

The Animal Landscape and Man Simulation System (ALMaSS) (Topping et al. 2003) is a simulation system of agent-based models developed over the past 20 years for evaluating the impact of management on organisms simulated in realistic landscapes. ALMaSS is focused on agricultural systems issues and has been used particularly to evaluate agri-environmental scheme implementation (e.g., Langhammer et al. 2017; Topping et al. 2019) and impact assessment of pesticides (e.g., Topping et al. 2018; Ziółkowska et al. 2021). To evaluate management impacts using ALMaSS, the species models are integrated into a highly detailed landscape simulation (Topping and Duan 2024a). ALMaSS landscape simulation has a resolution of the underlying topographical map of 1 m^2 and simulates daily farm and field management and plant growth.

Until now, all ALMaSS species models have been developed as agent-based models (Duan et al. 2022; Capela et al. 2024; Topping and Duan 2024b) simulating detailed individual behaviour and decision-making (e.g. Topping and Odderskaer 2004; Topping et al. 2010). Agent-based models use the local informational context and the model species' own condition to make decisions to further an agenda of development, movement, survival and reproduction. The result is an emergent population response, based on the action and interaction between these agents and between the agents and the environment. However, the agent-based approach is not applicable for certain species (e.g. aphids) due to a lack of detailed individual-based information, the huge numbers of individuals that can be widespread across a landscape or both. In addition, these species are often the prey for other species, which could be modelled as agents (e.g. aphids as prey for ladybirds). Thus, we need to model billions of individuals time efficiently since they need to be run simultaneously with other agent-based models. Hence, we had to develop a new ALMaSS population management approach to include these species in the simulation system.

The key design objectives for the new approach were as follows:

- Be capable of representing the entire population for the simulated landscape;
- Include the detailed landscape representation and be able to interact with landscape, field and crop management;
- Include spatial dynamics of the species;
- Accurately represent the population development of the species;
- Run efficiently on landscapes of standard 10 × 10 km.

The proposed subpopulation model integrates three types of model: stage-structured population models, spatially-structured population models and landscape simulation. Previous models typically use one of these approaches. The Leslie Matrix is one of the best-known stage-structured population models (Leslie 1945). While the Leslie Matrix model specifically focuses on age-structured populations, stage-structured models encompass a broader range of characteristics, including life stages that may not be directly tied to chronological age. In stage-structured population models, individuals are classified into different stages, based on specific characteristics, such as size, developmental stage or reproductive status. Each stage is associated with its own set of transition probabilities, representing the likelihood of individuals moving from one stage to another. This type of model is particularly useful when age alone does not fully capture the variations in the life history of a species. Using this form of model as a base allows a large degree of flexibility in defining our subpopulation model methods.

Bommarco et al. (2007) used a lattice model and individual dispersal of carabids to simulate an aphid/carabid system and Parry et al. (2006) used a detailed individual-based model to simulate aphids in relatively simple landscapes. The closest model to the proposed model in concept is probably the DYMEX model, which also links a more traditional population approach to spatial dynamics through dispersal (Parry et al. 2011). Other approaches to modelling populations, such as aphids, include the use of a more traditional mathematical simulation approach, in some cases based on spatially distributed observations (e.g. Knudsen and Schotzko 1999; Winder et al. 2001; Dutta et al. 2008). Similarly, Damgaard et al. (2020) used a stochastic aphid population growth model

and fitted the model to an empirical spatial time series of aphid population data using a Bayesian hierarchical fitting procedure to create predictions of aphid population in time and space. In the case of Brown et al. (2021), a multi-patch model was developed to look at the dynamics of aphids and ladybird predators, using a differential equation approach expanded to include details of the attraction of the habitats and details of dispersal.

Our approach adapts a standard stage-structured population model to create a spatially dynamic model interacting with ALMaSS landscape models. We believe the combination created here to be unique because of the integration with the detailed landscape modelling. The stage-structured population model by Lefkovitch (1965) described population growth by breaking the population down into stages (often physiological life stages with an individual life history (Botsford et al. 2019)) and tracking the development of the population by describing rates of change from one stage to another. We used this basic concept for the ALMaSS subpopulation model to create a general subpopulation model that can be easily adapted to represent different species with different life cycles and types of behaviour. To implement the spatial dynamics, we split the entire population in the simulated landscape into a grid of stage-structured subpopulations, each potentially linked to all others by dispersal. For each life stage, the abundance at different ages is tracked daily for every grid. Within the subpopulation, the standard population processes of reproduction, mortality, development and dispersal are represented by separate, but linked modules within the model. The model can be tuned to represent a wide range of species by specifying the values of parameters for vital rates and altering the structure of the table describing the stage-structured population.

This paper describes and documents the foundational ALMaSS subpopulation model framework, specifically designed to model species with large population sizes. It is important to note that the collection of biological information and parameterisation for specific species are beyond the scope of this paper. Instead, we demonstrate the model's design objectives and general applicability through a hypothetical generic species and provide examples of aphid models developed using this subpopulation model framework.

Methods

The implementation of the general subpopulation model within ALMaSS is achieved through two C++ classes, "Subpopulation" and "Subpopulation_Population_Manager". The "Subpopulation" class represents a single subpopulation within a grid cell and an object of this class is used to simulate the stage-structured subpopulation in each grid cell (with a default size of 10 m × 10 m). The "Subpopulation_Population_Manager" class is an administrative class that manages the entire population across the simulated landscape, which comprises all "Subpopulation" objects and provides input/output functions for the subpopulation model.

These two classes contain all the necessary code to implement the subpopulation model framework, which can be based on modelling specific species in the real world, such as aphids (Thomsen et al. 2024). The proposed model is integrated with the standard ALMaSS landscape simulation model, which provides the model species with dynamic, landscape-specific information as the simulation progresses. This integration allows the species to interact realistically with its environment throughout the simulation.

This paper's description of the general subpopulation model assumes a hypothetical generic species (not based on any real-world species, purely for illustration purposes) with five life stages: egg, larva, pupa, winged adult and wingless adult. Adults are only females and they are all mated and are capable of laying eggs. The winged adults are the only life stage capable of long-distance dispersal through flight, which they can undertake once in their lifetime. Since their flight is influenced by wind direction and speed, it is considered partly passive transportation. After flying, they become wingless adults. Local movement occurs by shifting a certain proportion of the subpopulation to one of the adjacent cells surrounding the current location. Thus, the distance they can cover in a single short-distance dispersal event depends on the size or dimensions of these cells. Eggs, larvae and pupae are immobile within their grid cells. When applying the general subpopulation model to a specific species, biological details such as life stages (as shown in Table 1) must be incorporated into the proposed model to accurately reflect the species' characteristics.

The Subpopulation_Population_Manager class

This class represents the entire population across the simulated landscape as a collection of subpopulations. For simplicity, the "Subpopulation_Population_Manager" class will be referred to as the manager class throughout the rest of this paper. Subpopulations are defined by applying a regular grid (default size of 10 m × 10 m) over the simulated landscape. For each grid cell, an object of the "Subpopulation" class is created and managed daily by the manager class. For example, a 10 km × 10 km landscape with a 10 m × 10 m grid size would result in 1 million "Subpopulation" objects. Fig. 1 illustrates the running flow of the manager class, showing initialisation and the daily time-step loop, which continues until the simulation concludes.

When an instance of the manager class is created, initialisation is triggered. During this process, the manager loads all parameters and host information (see Table 1) and divides the landscape into grid cells. Each grid cell is assigned an instance of the "Subpopulation" class, with the suitability of each cell for the species calculated, based on the landscape composition within that cell. A key procedure during initialisation is the pre-calculation of the long-distance dispersal landing mask (see "Long-distance dispersal pre-calculation"). After initialisation, the manager enters the main simulation loop, advancing in one-day time steps and repeating until the simulation is complete.

DoFirst function

The first part of the main simulation loop is to call the DoFirst function of the manager class. Here, the development season is updated first. Some species, for example, black bean aphids, switch plant hosts on which they have different life cycles. Based on this host switch, the development season is defined. If the model species do not have this behaviour, the development season will not change. Afterwards, the common part of the mortality rate depending on the environmental factors (e.g. temperature), which is identical for the whole simu-

The running flow in the SupPopulation_Population_Managerclass.

```
Initalisation
    Read the development file
    Read the host file
    Precalculate the landing masks
    Creat the SubPopulation objects
repeat
    DoFirst()
        updateDevelopmentSeason()
        updateMortalityArray()
        relocatePopulation()
    Loop all SubPopulation objects in parallel
        BeginStep()
            Nothing in this base class
    Loop all SubPopulation objects in parallel
        Step()
            calPopuDensity()
            calSuitability()
            doMortality()
            doMovement()
            doDevelopment()
            doReproduction()
    Loop all SubPopulation objects in parallel
        EndStep()
            Nothing in this base class
    DoLast()
        doDevelopment()
        doSpeciesLastThing()
        addNewDay()
        updateWholePopulation()
        subpopuBaseOutputProbe()
until the end of the simulation
```

Figure 1. Running flow of the Subpopulation_Population_Manager class.

lated landscape at each time step, is also calculated. The flying animals will be distributed using the landing mask, based on the wind speed and wind direction for that day in this DoFirst function as well.

BeginStep, Step, EndStep functions

After the preparatory steps in the DoFirst function, a sequence of three loops for all the "Subpopulation" objects is initiated to call functions from the "Subpopulation" object. The first loop calls the BeginStep function for all "Subpopulation" objects. Although this function is typically empty in the base class, it can be overridden in a derived model class if needed. The purpose of BeginStep in a derived class is to prepare the "Subpopulation" objects for the next loop, where their Step function is called.

The second loop runs immediately after the BeginStep loop, iterating over all "Subpopulation" objects to execute their Step function. The Step function encompasses behaviour related to population density, habitat suitability, individual development, reproduction, mortality and dispersal. Once the Step

function has been executed for all "Subpopulation" objects, the third loop calls the EndStep function for each one to conclude the day's activities. Like BeginStep, the EndStep function is empty in the general subpopulation class, but can be overridden in a derived class if specific actions are required at the end of the time step for the real model species.

These three loops are parallelised using OpenMP (Chandra 2001) at the thread level, which significantly enhances performance. Given the potentially large number of grids (up to 1 million in standard configurations), processing grids sequentially would be extremely time-consuming. Parallelisation is, therefore, a key design feature that enables the model to run efficiently, even with a large number of "Subpopulation" objects.

DoLast function

After cycling through all "Subpopulation" objects, common calculations shared by all the "Subpopulation" objects are executed within the DoLast function. These calculations yield landscape-wide results used by all subpopulations. The doDevelopment function calculates the accumulated day degrees, which will be detailed in "Development calculation (doDevelopment function)". The doSpeciesLastThing function, while empty in this general subpopulation model, offers a placeholder for potential species-specific overrides. To advance simulation time, addNewDay increments to the day counter. Finally, updateWholePopulation and subpopuBaseOutputProbe provide population recording, the former aggregating totals and the latter recording daily numbers.

Other functions

Alongside core calculations, the manager provides four additional functions for "Subpopulation" objects. These are the next life stage calculation, reproduction calculation and functions linked to the development calculation (supplyOldEnoughIndex and supplyAgeInDay) (see Fig. 1). These are described in the relevant sections below.

Initialisation

The first operation in the initialisation loads model parameters related to species development and host plants, as illustrated with the hypothetical generic species in Table 1. Grid cell suitability is determined by the types of habitats and plants present, necessitating a list of suitable host plants. These host lists, stored in an external text file, are divided into summer and winter categories, as some species utilise different vegetation depending on the season. If a species does not change hosts seasonally, only the summer list is used. For each season, two lists are created: one for habitat types (e.g. forests, buildings, gardens) and another for crops. Arable fields are excluded from the habitat list and, instead, represented in the crop list, as their vegetation is dynamically determined by the farming simulation in ALMaSS.

During initialisation, life stages capable of flight are added to a vector, with corresponding stages for post-landing stored in another vector. This accounts for species that may shed their wings after landing. If the flying and landing

stages are identical, the flying ability is retained until death or until a single flight has occurred.

Additionally, a "Subpopulation" object is created for each grid cell. A suitability variable within each "Subpopulation" object is initialised, based on the habitat or crop type at the grid's centre. If this type matches any in the host list, the suitability is set to 1; otherwise, it is set to 0. Upon creating the "Subpopulation" object, a predefined initial population of the model species at a specific life stage is added, provided the suitability value is greater than 0. In this hypothetical generic species, 1,000 eggs are introduced. This process is carried out in the initialisePopulation function.

Update development season

This function controls the development season, winter or summer development. When the model species does not switch hosts, this function returns summer development as default, which means only summer (yearly) hosts are used. In this general subpopulation model, we assume there is only one development season. If there is more than one in the derived class, for example, with winter hosts, this function requires an override in the derived class for the model species.

Update mortality array

Since the environmental factors, for example, average temperature, wind speed and direction, are identical for the whole simulated landscape, the base mortality rate, only depending on environmental factors, is calculated and stored in the manager class. When the "Subpopulation" object needs this, it does not need to be recalculated.

In this general subpopulation model, we assume a base mortality rate of 0.01 for all life stages of the generic species. This function needs to be overridden in the derived class and will typically be linked to weather or other factors controlling a general 'background' mortality for the model species.

Long-distance dispersal pre-calculation

This function is used to pre-calculate the landing mask for the model species doing the long-distance dispersal for the sake of computing efficiency.

We assume the long-distance flying dispersal is influenced by the wind, but this can only happen when the wind speed is lower than a certain value. Long-distance flying is done using 2D landing masks, which are generated by the landing proportion 1D curve along the wind direction. Below are the key parameters with the default values in the parenthesis for the long-distance flying dispersal.

- **Maximum wind speed**, w_M (12 m/s): When the wind speed is larger than w_M, the model species will not attempt to fly;
- **Wind step size**, w_S (2 m/s): This is used to sample the wind speed to discrete ranges. Each range has four landing 2D masks in four wind directions;
- **Peak distance**, d_p (100 m): This is the distance where most of the flying ones can land along the wind direction when the wind speed is in the first range (0 to w_S);

- **Longest distance**, d_l (1000 m): This is the longest distance at which the flying ones can land along the wind direction when the wind speed is in the first range (0 – w_S);
- **Wind speed scale**, S_W (1.2): This is the scale used to increase the peak and longest distance when wind speed increases to one range from the previous range.

The landing curve along the wind direction

Two polynomial functions are used to generate a long tail landing curve along the wind direction, as shown in equation (1):

$$p(l) = \begin{cases} -l^2 + 2d_p l, & if\ l \leq d_p \\ \dfrac{d_p^2 (d_l - l)^2}{(d_l - d_p)^2}, & if\ l > d_p\ and\ l \leq d_l \end{cases} \quad (1)$$

where l is the distance from the landing position to the original position. In this way, the position at d_p will have the largest landing proportion compared to other positions. The dispersers can land at a maximum distance of d_l, beyond which no one can land ($p(l) = 0$). The landing curves calculated using the default parameter values can be found in Fig. 2. When the wind speed enters a new (higher) wind speed, range, d_p and d_l are multiplied by the wind speed scale (S_W =1.2 by default) for the calculation in equation (1).

2D landing mask

After obtaining the landing curve along the wind direction, we need to convert it to a 2D mask in order to land the dispersers with wings on a 2D landscape. Firstly, we rotate the landing curve 360 degrees through the original location.

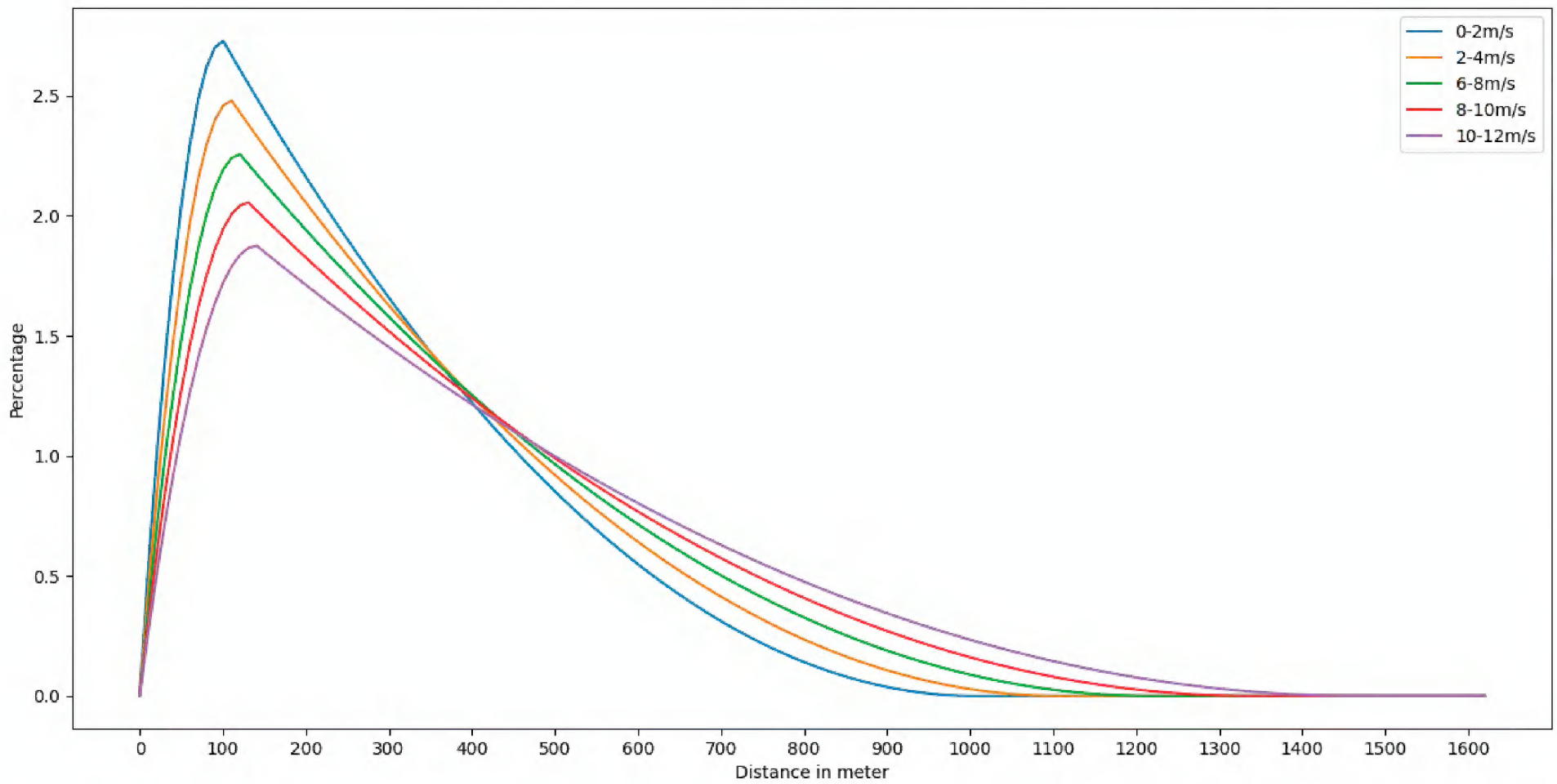

Figure 2. Landing curve examples calculated using equation (1) for five wind speed ranges. When the speed is larger, the model species can fly further away from their originations (denoted by 0 on the x-axis).

If we suppose that the original location is at (0,0), then the distance from each location denoted by (*x*,*y*) in a 2D landscape to the original location can be calculated using equation (2):

$$l = \sqrt{x^2 + y^2} \tag{2}$$

After equation (2), we can obtain the initial symmetric 2D landing mask to the source location using equation (3):

$$L'(x,y) = p(l) = p\left(\sqrt{x^2 + y^2}\right) \tag{3}$$

Next, we skew the landing mask, based on wind direction. We want more individuals to land downwind and none to land directly upwind. To achieve this, we use the cosine of the angle between the dispersal path (origin to destination) and the wind direction as a weight.

We suppose the wind direction to the source location is given by (*b*, −*a*) and the direction from the destination location to the source location is given by (*y*, −*x*). Then, the weight used to skew the initial landing mask can be calculated using equation (4):

$$w_c(x,y) = \left(\frac{by + ax}{\sqrt{x^2 + y^2}\sqrt{a^2 + b^2}}\right)^m \tag{4}$$

where *m* is an integer to control the landing range; when *m* is larger, more individuals will land along the wind direction. Afterwards, the 2D mask with the wind direction of (*b*, −*a*) can be calculated by equation (5) as exmapled in Fig. 3.

Four wind directions (N, S, W, E) are used in the proposed model. Each wind speed range has one landing mask for every wind direction. The landing masks are normalised so their sum equals 1, ensuring appropriate distribution during landing calculations. One example of the 2D mask with a wind direction for five wind speed ranges can be found in Fig. 3.

$$\mathrm{L(x,y)} = \begin{cases} L'(x,y)w_c(x,y), & if\ w_c(x,y) > 0 \\ 0, & else \end{cases} \tag{5}$$

The Subpopulation objects loop

All the “Subpopulation” objects are looped three times to run their BeginStep, Step and EndStep functions, respectively, which are done in parallel within each containing loop using OpenMP. The details of the Step function can be found in “The Subpopulation Class”, while the BeginStep and EndStep are empty and are placeholders for modelling real species.

Development calculation (doDevelopment function)

Since the whole simulated landscape window has the same environmental temperature, the degree-days used for the physiological age of the model species are calculated in the manager class at every time step, i.e. not individually in the “Subpopulation” class. Each life stage has its base development temperature (T^i_m, where *i* is the life stage number). For each simulated day, the degree days (DD^i) are calculated using equation (6), based on the daily average tempera-

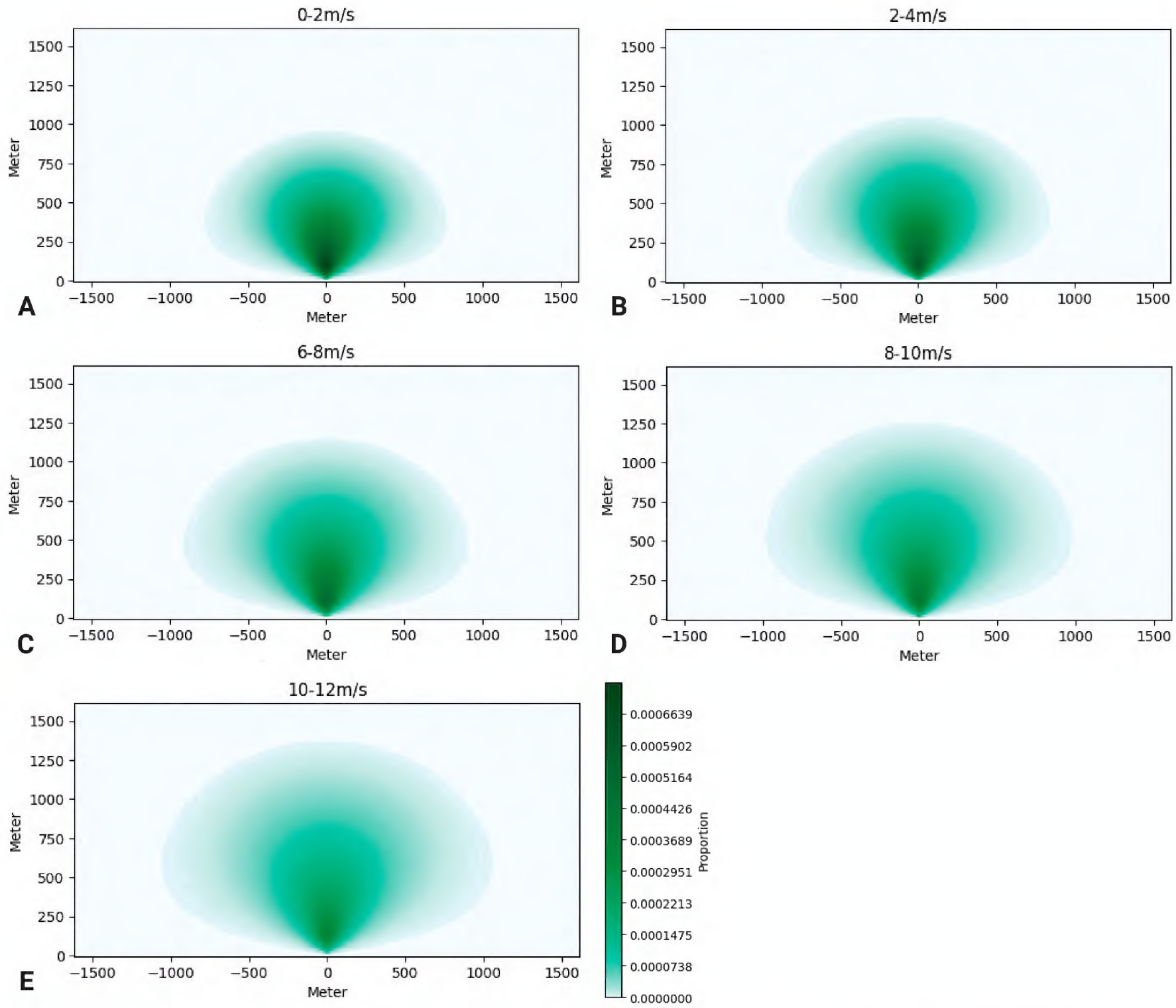

Figure 3. 2D landing mask examples with the colour bar for the landing proportions. The winged adult flies from the original (0, 0). The wind direction is along the y-axis (from bottom to top). The wind adults will not land at the locations where the flying direction is opposite to the wind direction. Further, when the wind speed is increasing, they can reach further locations along the wind direction. *m* in equation (4) is set to be 3 in these examples.

ture (T_a). When T_a is smaller than T^i_m, no degree days will be accumulated (i.e. $DD^i = 0$ when $T_a \leq T^i_m$):

$$DD^i = T_a - T^i_m \quad (6)$$

The model species accumulate day degrees by the creation of a 2D array, ***D***, of which the first dimension is the index of the life stage ($i = 0,\ldots,N_l - 1$), while the second is the calendar day counter ($j = 0,\ldots,N_c - 1$). Each element D_{ij} is the accumulated day degrees by adding the value calculated using equation (6) for life stage *i* which are born or turned to that life stage at calendar day represented by *j*. All the elements of ***D*** are initialised with 0 from the beginning of the simulations.

In addition to this 2D array, each life stage has two variables for the oldest (o_i) and the newest index (b_i) in the second dimension of ***D***. The oldest position o_i is where the oldest extant population is placed in ***D*** for life stage *i*. The newest position b_i is where the newborn/newly-developed population should be added for life stage *i*. From the beginning, both the oldest and newest indices start as 0. At the end of each day:

1. The newly-accumulated day degrees will be added to D_{ij}, for the positions that meet the equation (7):

$$j \in \begin{cases} [o_i, b_i], & if\ b_i \geq o_i \\ [0, b_i] \cup [o_i, N_c - 1], & if\ b_i < o_i \end{cases} \quad (7)$$

2. The newest position is added by 1 to move to the next day. If the newest position is larger than or equal to N_c − 1 after the addition, it will be set to 0.
3. For each life stage, the accumulated degree days pointed to by the oldest index are compared with the maximum development degree days for each life stage (denoted by D^i_M); if it is larger than or equal to the maximum one, the oldest index will be inserted into the ready-to-next-life-stage array. The accumulated day degrees in ***D*** indexed by the oldest index will be reset to zero. Afterwards, the oldest index will be incremented. If it is larger than or equal to N_c − 1, the oldest index will be set to 0. This check and addition will continue until the accumulated degree days at the oldest index are less than the maximum development degree days.

There are two functions linked to development calculation: 1) supplyOldEnoughIndex function returns the ready-to-next-life-stage array, which will be used by the "Subpopulation" class; 2) supplyAgeInDay function returns the age in calendar days for the given life stage and index in the second dimension of ***D***. To calculate the age of the population, the accumulated degree days at D_{ij} in calendar day (a^c_{ij}), the equation (8) is used:

$$a^c_{ij} = \begin{cases} b_i - j, & if\ b_i \geq o_i\ or\ b_i < o_i \wedge b_i > j \\ N_c - j + b_i, & if\ b_i < o_i \wedge b_i \leq j \end{cases} \quad (8)$$

Next life stage calculation

This function calculates whether the model species at one life stage has accumulated enough degree days to transition to the next life stage. This function needs to be overridden in the derived class to accurately describe the species' biological information to be modelled. The general "Subpopulation" class example used here has five life stages, namely, egg (0), larva (1), pupa (2), winged adult (3) and unwinged adult (4). The number in the brackets represents the life stage, while −1 means death. If the current life stage is l_c, the next life stage (l_n) can be calculated using equation (9). The winged and wingless proportion is controlled by the population density, as described in "*Calculation of population density (calPopuDensity function)*":

$$l_n = \begin{cases} -l_c + 1 & l_c < 2 \\ -1 & l_c \geq 3 \\ 3\ or\ 4 & l_c = 2 \end{cases} \quad (9)$$

Offspring calculation

This function calculates the number of offspring and the life stage number of the offspring for the given life stage. It is called by the reproduction function in the "Subpopulation" class. In this base class, only the adults can give birth to one egg each day. An override is always required for a derived class — for example, species like aphids that can lay eggs or give birth to nymphs.

The Subpopulation Class

An instance of the "Subpopulation" class is created within each grid cell to track the population dynamics specific to that cell. The primary function in the "Subpopulation" class is the Step function, which is called by the manager class during its loop for all the "Subpopulation" objects. The flowchart for the Step function is illustrated in Fig. 1. The BeginStep and EndStep functions are empty in this base class and are reserved for derived classes that may require additional operations before or after executing the Step function. The specific functions within the Step function are detailed in the subsections below.

Calculation of population density (calPopuDensity function)

The calPopuDensity function calculates the population density for the grid cell. This function must be overridden in the derived class. In the base "Subpopulation" class, population density is calculated using the suitable host green biomass available in the cell. If this density exceeds a user-defined threshold, winged adults are developed, prompting long-distance dispersal.

Calculation of suitability (calSuitability function)

The calSuitability function recalculates host suitability daily to account for crop rotations in arable areas. If a crop in the cell is not on the host list, suitability is set to 0 and the doMortality function subsequently removes the subpopulation from the grid. Suitability is reset to 1 once a crop from the host list is present again.

Mortality (doMortality function)

Mortality rates for each life stage are managed through user-defined parameters or calculation functions. These require an override of the function of calCellRelatedMortalityWeight in the derived classes. The calCellRelatedMortalityWeight function is used to calculate the marginal mortality related to the grid, which is empty in the base class. In the doMortality function, a certain number of individuals in each grid at each life stage will die according to their mortality rate, using the grid cell's suitability value as weighting. When the suitability value is 0, all the populations in the grid will be killed by this function. The doMortality function does NOT need an override in derived classes.

Movement (doMovement function)

The local movement (short-distance movement compared to the long-distance movement in the manager) is implemented here. Short-distance movement occurs independently of long-distance flight. Further, we assume that only a certain proportion of the individuals can move and they can only move to one of the eight adjacent cells, which is chosen randomly. After the target cell is chosen, they only move to it if the destination's suitability (see "*Calculation of population density (calPopuDensity function)*") is greater than zero and the destination's population density (see "*Calculation of population density (calPopuDensity*

function)") is lower than the departure cell. This movement is typically used for adults, but can applied to any life stage if they are capable of moving from one place to another.

Development (doDevelopment function)

The development of the subpopulation is implemented, based on the development array in the manager class (see "Development calculation (doDevelopment function)" in the manager). In the "Subpopulation" class, a 2-D array ***P*** is used to track the subpopulation of the model species in each life stage at different ages in a grid cell. The first dimension of ***P*** is to index the life stage. The second dimension of ***P*** is to index the calendar days. Importantly, the second index does not represent absolute age. Instead, it groups animals together, based on the day they were born or transitioned to their current life stage, i.e. p_{ij} is the number of modelled species at life stage i and having the age indexed by j. To obtain the age in calendar days for these p_{ij} modelled species, supplyAgeInDay function from the manager class is used. Further, all these p_{ij} modelled species have accumulated the same D_{ij} degree days which is stored in the manager class.

At each time step, the "Subpopulation" class calls the supplyNextLifeStage function in the manager for every life stage to obtain the index (j) pointing to the animals that are ready to move to the next life stage. The animal number represented by p_{ij}, indexed by j for life stage i, will be added to the position indexed by the newest position variable from the manager class for the next life stage, effectively moving individuals from the last index j at life stage i to the newest index in the next life stage. If the next life stage index is −1, the population will be removed from the array of ***P*** since −1 indicates death.

Reproduction (doReproduction function)

The doReproduction function handles reproduction for eligible life stages, with the calOffspringStage function from the manager class returning the number and life stage of offspring. In the base subpopulation class, it is assumed for simplicity that only adults lay one egg per day. The doReproduction function must be overridden in derived classes to customise reproductive behaviour for real model species, based on its our own biological information.

Results

Hypothetic generic species

The proposed model framework was implemented to seamlessly interact with the existing ALMaSS landscape model. To illustrate its application, we conducted a five-year simulation on a 10 km × 10 km Danish landscape using the hypothetical generic species parameterisation, as shown in Table 1. These parameters were randomly assigned, serving solely to demonstrate the model's functionality without relying on real biological data. The biological data will be required to model a real species, such as aphids (Thomsen et al. 2024), to obtain realistic parameter values for the parameters listed in Table 1.

Table 1. Model parameters with values used in the hypothetic generic species. TOC is an enumeration of the types of crops present in ALMaSS. TOLE is an enumeration listing the types of landscape elements (habitats) in ALMaSS. Please note that these values do not contain any real biological information since the hypothetic generic species is not a species in the real world. This is only for illustration purposes.

Name	Unit	Value	Symbol	Note
The number of life stages	–	5	N_l	egg (0), larva (1), pupa (2), winged adult (3), wingless adult (4)
The maximum lifetime	Calendar day	200	N_c	The maximum alive days amongst all the life stages
Maximum degree days	Degree days	80, 90, 100, 213, 213	D^i_M	The maximum degree days for each life stage are denoted by *i*.
Base development temperature	°C	10, 5.5, 5, 5, 5	T^i_m	The lower temperature that the model species can accumulate day degrees for each life stage is denoted by v *i*.
The size of the grid	metre	10 × 10	–	Width and height
The flying array		3	–	Life stages that can fly
The landing array		4	–	Life stages after the flying ones landed
Winter host list (habitat)	–	–	–	TOLE, is not used in the generic species
Winter host list (crop)	–	–	–	TOC, is not used in the generic species
Summer host list (habitat)	–	tole_RoadsideVerge tole_BeetleBank tole_RoadsideSlope tole_Hedges tole_FieldBoundary tole_NaturalGrassDry tole_WoodlandMargin tole_Vildtager tole_NaturalFarmGrass tole_OPermPasture tole_OPermPastureLowYield tole_NaturalGrassDry tole_PermPastureLowYield	–	TOLE
Summer host list (crop)	–	Spring Barley	–	TOC

Fig. 4 illustrates the resulting population dynamics in the fifth year, demonstrating the model's success in capturing annual population cycles and handling billions of populations. The number of eggs at the end of the fifth year differs from that at the beginning of the year due to the influence of landscape dynamics, including weather conditions and crop variations, on population dynamics. While the population may reach relative stability, these environmental factors continue to impact fluctuations. This demonstrates that our model effectively integrates with ALMaSS dynamic landscape models. Additionally, we evaluated runtime performance under different threading number configurations (Table 2). As expected, the use of multi-threading significantly reduced simulation time, though gains diminished with the addition of further threads.

Table 2. Running time with different thread numbers for a five-year simulation.

Number of Threads	1	5	10	20
Running time in second	1472	748	638	615
Relative to running time with one thread	1	0.508	0.433	0.417

Real species – four aphid species

Based on the proposed subpopulation model, four aphid species (*Aphis fabae*, *Acyrthosiphon pisum*, *Myzus persicae* and *Sitbion avenae*) were implemented within the ALMaSS simulation environment. Please refer to Thomsen et al. (2024) for the details of these aphid models. A ladybird model was developed using the

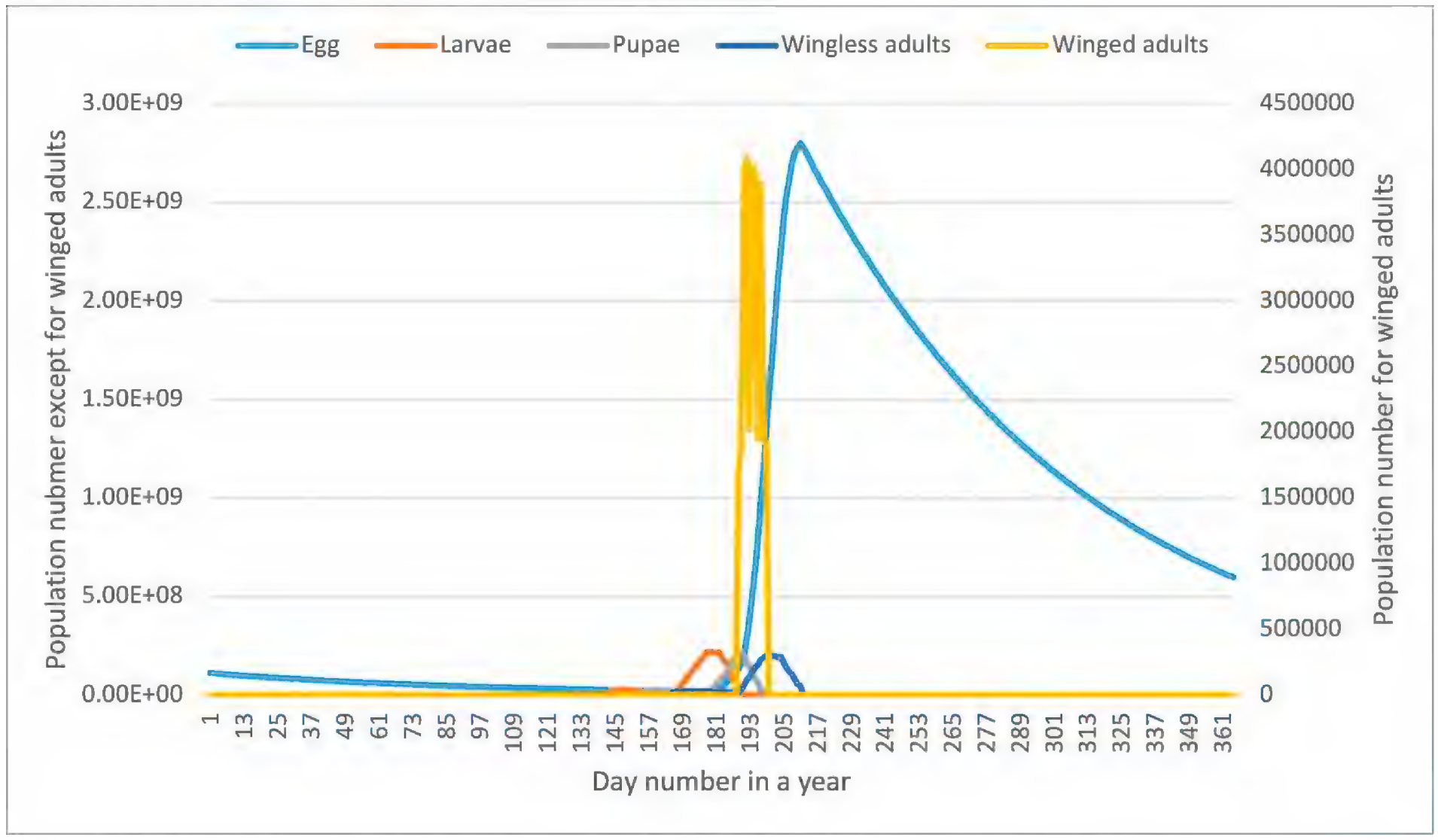

Figure 4. Population dynamics of the generic species in the fifth year from a five-year simulation. Please note the number of winged adults uses a separated y-axis to make it visible due to its smaller value compared to other life stages.

agent-based method to simulate predator-prey interactions. Our implementation allows ladybirds to dynamically interact with the four aphid species within the ALMaSS landscape to prey on the aphids. The proposed subpopulation method was able to handle billions of modelled aphids of four species combined in a single 30-year simulation running simultaneously with the agent-based ladybird model. Fig. 5 presents a heatmap time series visualisation, demonstrating the complex population dynamics and spatial patterns that emerge from the simulated aphid-ladybird interactions. The shading in the images represents the aphid populations over time. Initially, as shown in the first image, aphids are concentrated in natural habitat areas. As crops grow, the aphids migrate into crop areas through long-distance dispersal, which is depicted by the increasingly brighter areas in subsequent images, indicating aphid reproduction. Ladybirds, represented by purple dots, track the aphid populations and begin to reproduce in response to aphid abundance. The reproduction of ladybirds is visible through yellow and green dots, indicating the life stages of the predator. This figure showcases the spatial and temporal complexity of interactions between aphids and their predator, the ladybird, within a dynamic landscape. It highlights both aphid population growth and predator responses, illustrating how these populations fluctuate and interact over time along with the landscape dynamics.

Discussion

In this paper, we introduced a general subpopulation model framework designed to handle species with large populations. This framework operates by tracking the population development of species across a simulated landscape, recording the number of individuals at various life stages and ages within each

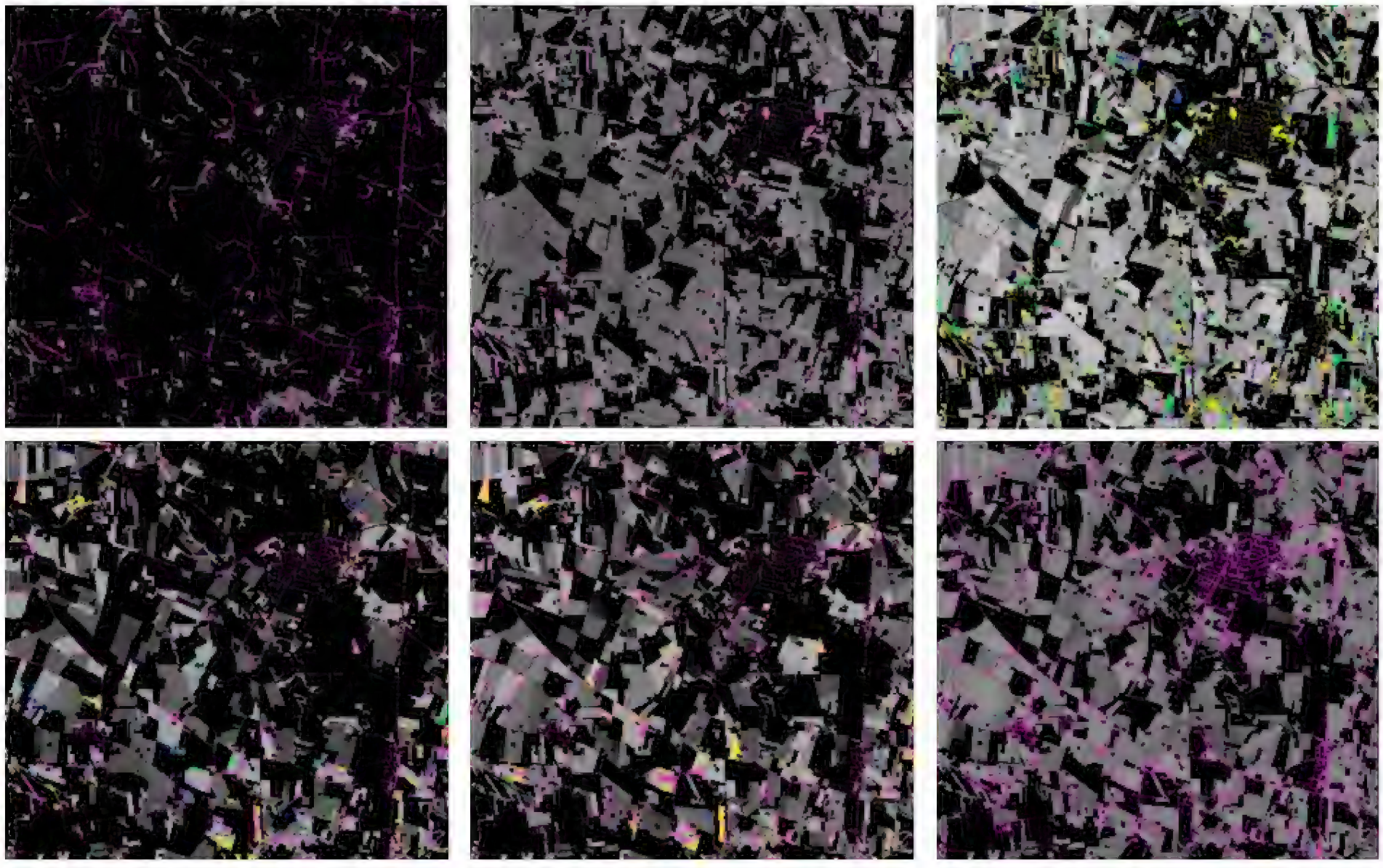

Figure 5. Subpopulation-based aphid models with a ladybird agent-based model as predators, simulated in a 3 km by 3 km Danish landscape over various time intervals. Shading represents aphid density: black indicates no aphids, while brighter shades signify higher densities. Coloured dots represent ladybirds at different life stages.

grid cell. By aggregating these local subpopulations, the model can represent the entire population across the landscape, providing a comprehensive view of population dynamics.

Our model differs from traditional agent-based models by operating on subpopulations belonging to different non-overlapping grid cells in the landscape rather than individuals, using the number of individuals at specific life stages and at different ages as the minimal unit. While this approach offers less granularity than modelling each animal as an individual agent, it still achieves a fine resolution by allowing stage-specific age distinctions (e.g. all 12-day-old larvae). This balance enables us to capture detailed population dynamics without the computational overhead of full agent-based modelling, making it particularly suited for large-scale simulations.

By integrating with existing ALMaSS landscape models, our subpopulation framework can incorporate dynamic environmental factors, such as vegetation growth and agricultural practices. This connection allows the model species to interact realistically with their landscape, responding to factors like field management decisions (e.g. insecticide treatments) and seasonal vegetation changes. As demonstrated in Fig. 5, aphids modelled using the proposed subpopulation model framework appear only in suitable habitats and migrate into crop areas as they become available, reflecting how the model accounts for both spatial and temporal landscape variations.

Our model links subpopulations in different cells to capture spatial population structure through dispersal mechanisms rather than relying on traditional metapopulation or patch dynamic models. Dispersal mechanisms include

long-distance flights and local movement between grids, facilitating accurate spatial dynamics across the landscape. The landscape model provides a dynamic context, affecting subpopulation behaviour through factors like wind speed and direction. This approach allows for the simulation of complex dispersal scenarios, where subpopulations can respond adaptively to changing environmental conditions.

Our multi-threaded approach enables efficient simulations, even for complex scenarios. For instance, we successfully ran a 30-year simulation involving four aphid species and one ladybird species running simultaneously across a 10 km-by-10 km landscape in just 20 hours using 30 threads. This performance demonstrates the model's capability to handle extensive simulations, while maintaining computational efficiency.

While the model's conceptual structure is relatively straightforward, its implementation requires a solid understanding of the ALMaSS system and its programming environment, reflecting a trade-off between complexity and flexibility. Despite this, the framework is sufficiently generic to model a wide range of species, making it a versatile tool for future ecological modelling. Provided as a set of C++ classes, this model can be directly integrated into the ALMaSS software environment or adapted to other modelling systems capable of supplying the necessary data. This flexibility makes it a valuable resource for modellers seeking to explore large-scale population dynamics in complex landscapes.

Conclusions

This paper presents a novel base model designed to serve as a foundational framework for modelling species with large populations. This subpopulation model provides a flexible tool for developing species-specific simulations by focusing on population dynamics within detailed simulated landscapes. Our approach enables a more accurate understanding of types of population behaviour without requiring individual-level modelling by integrating elements from stage-structured and spatially structured population models with landscape simulation. Unlike previous models that often rely on a single approach, this framework leverages the strengths of multiple methodologies. It combines detailed landscape modelling with nuanced subpopulation dynamics by segmenting landscapes into non-overlapping grids, allowing for the simulation of large-scale systems involving billions of individuals. Already successfully applied to simulate aphid populations, this model serves as a robust foundation for building new, species-specific models within ALMaSS or other simulation projects. These models can be tailored to meet the unique requirements of different species.

Acknowledgements

We would like to thank Sara Nafisi for her assistance with the figures in this paper.

Additional information

Conflict of interest

The authors have declared that no competing interests exist.

Ethical statement

No ethical statement was reported.

Funding

This work was supported by DeiC National HPC (UCloud, GenomeDK, LUMI) (g.a. DeiC-AU-N1-000025, DeiC-AU-N2-2023015 and DeiC-AU-L5-0011) and by the EcoStack project, funded by the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement no. 773554

Author contributions

Both authors have contributed equally.

Author ORCIDs

Xiaodong Duan https://orcid.org/0000-0003-2345-4155
Christopher J. Topping https://orcid.org/0000-0003-0874-7603

Data availability

All of the data that support the findings of this study are available in the main text.

References

Bommarco R, Firle SO, Ekbom B (2007) Outbreak suppression by predators depends on spatial distribution of prey. Ecological Modelling 201: 163–170. https://doi.org/10.1016/j.ecolmodel.2006.09.012

Botsford LW, White JW, Hastings A (2019) Population Dynamics for Conservation. Oxford University Press. https://doi.org/10.1093/oso/9780198758365.001.0001

Brown D, Bruder A, Kummel M (2021) Endogenous spatial heterogeneity in a multi-patch predator-prey system: insights from a field-parameterized model. Theoretical Ecology 14: 107–122. https://doi.org/10.1007/s12080-020-00483-6

Capela N, Duan X, Ziółkowska EM, Topping CJ (2024) Modelling foraging strategies of honey bees as agents in a dynamic landscape representation. Food and Ecological Systems Modelling Journal 5: e99103. https://doi.org/10.3897/fmj.5.99103

Chandra R (2001) Parallel programming in OpenMP. Morgan kaufmann, 240 pp.

Damgaard C, Bruus M, Axelsen JA (2020) The effect of spatial variation for predicting aphid outbreaks. Journal of Applied Entomology 144: 263–269. https://doi.org/10.1111/jen.12724

Duan X, Wallis D, Hatjina F, Simon-Delso N, Bruun Jensen A, Topping CJ (2022) ApisRAM Formal Model Description. EFSA Supporting Publications 19: 7184E. https://doi.org/10.2903/sp.efsa.2022.EN-7184

Dutta S, Bhattacharya BK, Rajak DR, Chattopadhyay C, Dadhwal VK, Patel NK, Parihar JS, Verma RS (2008) Modelling regional level spatial distribution of aphid (*Lipaphis erysimi*) growth in Indian mustard using satellite-based remote sensing data. International Journal of Pest Management 54: 51–62. https://doi.org/10.1080/09670870701472314

Knudsen GR, Schotzko DJ (1999) Spatial simulation of epizootics caused by *Beauveria bassiana* in Russian wheat aphid populations. Biological Control 16: 318–326. https://doi.org/10.1006/bcon.1999.0713

Langhammer M, Grimm V, Puetz S, Topping CJ (2017) A modelling approach to evaluating the effectiveness of Ecological Focus Areas: The case of the European brown hare. Land Use Policy 61: 63–79. https://doi.org/10.1016/j.landusepol.2016.11.004

Lefkovitch LP (1965) The Study of Population Growth in Organisms Grouped by Stages. Biometrics 21: 1–18. https://doi.org/10.2307/2528348

Leslie PH (1945) On the use of matrices in certain population mathematics. Biometrika 33: 183–212. https://doi.org/10.2307/2332297

Parry HR, Evans AJ, Morgan D (2006) Aphid population response to agricultural landscape change: A spatially explicit, indlividual-based model. Ecological Modelling 199: 451–463. https://doi.org/10.1016/j.ecolmodel.2006.01.006

Parry HR, Aurambout JP, Kriticos DJ (2011) Having your cake and eating it: A modelling framework to combine process-based population dynamics and dispersal simulation. In: 19th International Congress on Modelling and Simulation (MODSIM). Perth, Australia, 2535–2541.

Thomsen P, Duan X, Topping CJ (2024) Formal model for *Acyrthosiphon pisum*, *Aphis fabae*, *Myzus persicae* and *Sitobion avenae* using the ALMaSS subpopulation approach. Food and Ecological Systems Modelling Journal 5: e123747. https://doi.org/10.3897/fmj.5.123747

Topping CJ, Duan X (2024a) ALMaSS Landscape and Farming Simulation: software classes and methods. Food and Ecological Systems Modelling Journal 5: e121215. https://doi.org/10.3897/fmj.5.121215

Topping CJ, Duan X (2024b) Managing large and complex population operations with agent-based models: The ALMaSS Population_Manager. Food and Ecological Systems Modelling Journal 5: e117593. https://doi.org/10.3897/fmj.5.117593

Topping C, Odderskaer P (2004) Modeling the influence of temporal and spatial factors on the assessment of impacts of pesticides on skylarks. Environmental Toxicology and Chemistry 23: 509–520. https://doi.org/10.1897/02-524a

Topping C, Hansen T, Jensen T, Jepsen J, Nikolajsen F, Odderskaer P (2003) ALMaSS, an agent-based model for animals in temperate European landscapes. Ecological Modelling 167: 65–82. https://doi.org/10.1016/S0304-3800(03)00173-X

Topping CJ, Hoye TT, Olesen CR (2010) Opening the black box-Development, testing and documentation of a mechanistically rich agent-based model. Ecological Modelling 221: 245–255. https://doi.org/10.1016/j.ecolmodel.2009.09.014

Topping CJ, Dalby L, Skov F (2018) Developing spatio-temporal models for landscape-scale pesticide ERA. Copenhagen, Denmark, 146 pp.

Topping CJ, Dalby L, Valdez JW (2019) Landscape-scale simulations as a tool in multi-criteria decision making to support agri-environment schemes. Agricultural Systems 176: 102671. https://doi.org/10.1016/j.agsy.2019.102671

Winder L, Alexander CJ, Holland JM, Woolley C, Perry JN (2001) Modelling the dynamic spatio-temporal response of predators to transient prey patches in the field. Ecology Letters 4: 568–576. https://doi.org/10.1046/j.1461-0248.2001.00269.x

Ziółkowska E, Topping CJ, Bednarska AJ, Laskowski R (2021) Supporting non-target arthropods in agroecosystems: Modelling effects of insecticides and landscape structure on carabids in agricultural landscapes. Science of the Total Environment 774: 145746. https://doi.org/10.1016/j.scitotenv.2021.145746

Appendix 1

The source code and the running directory are publicly available at:

https://gitlab.com/ALMaSS/almass_methodology.git

The code documentation for the subpopulation model is publicly available at:

https://almass.gitlab.io/almass_methodology/Subpopulation.pdf